Action in Chains: A Chains Model for Action Localization and Classification

Gilad Sharir
KU Leuven ESAT/PSI, iMinds

Tinne Tuytelaars
KU Leuven ESAT/PSI, iMinds

Abstract

In this paper we present a method for action classification in videos using trajectory features. The novelty of our approach is in formulating the problem of simultaneous detection and localization as a probabilistic chains model. In our formulation, chains are sets of regions in the video that are connected based on their joint probabilities. We describe our approach for connecting subvolumes in the video into chains, and using them as spatio-temporal detectors for actions. Our approach allows the detection and localization of multiple actions occurring simultaneously or at different locations in a single video. We test the performance of our method on two challenging action recognition datasets, and compare to state of the art methods.

1. Introduction

Action classification in videos has been a long term goal of computer vision research. The research and methods developed for this task can be roughly divided into two main categories. Discriminative models aim at classifying actions by learning a direct map from test samples to the set of training examples and action labels. Given a new test sample of an action, the discriminative model assigns the sample to a certain class, and labels the sample according to the class label. Examples of discriminative models are SVM classifiers, which have been used extensively for action classification (e.g. [12, 14]).

Generative models on the other hand use the training samples to estimate a statistical model which best fits the data. Test samples are classified based on their consistency with the pre-trained model. Some examples of generative models for action recognition can be found in [13] and [6], which pose the problem of action classification as a latent topic model.

The generative models have the advantage of being able to describe data over multiple sources, and allow for appearance differences, yet they suffer from over generalizing and therefore typically achieve lower classification accuracy than discriminative models.

Our contribution is in proposing a detection and classification model that benefits from the classification accuracy of a discriminative model, and the generalizing ability of a generative model to simultaneously classify and localize actions within the video.

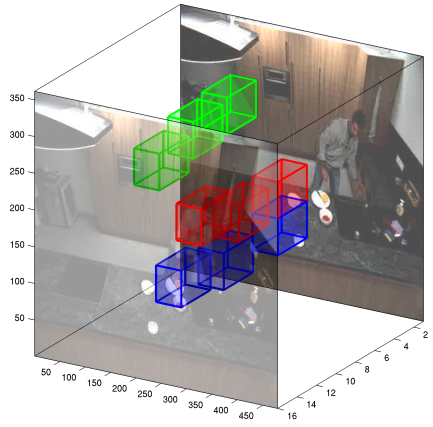

Figure 1. We detect and classify actions as spatio-temporal chains across the video. The different chains are scored according to the classification confidence and temporal coherence (see text). Chains on the background (green) or on uninformative parts of the foreground (red) are scored lower than chains on the disciminative parts of the object (blue).

Our model combines the generative and discriminative models in a way that benefits from the advantages of both approaches. As a generative model, we learn a probabilistic representation of each action class, by estimating the probability of co-occurrence of neighboring regions of a video. Our discriminative model is able to accurately classify these regions based on the features they contain. We then extend these regions detected by the discriminative model into longer *chains* that correspond to the temporal patterns learned by the generative model. An example of probabilistic chain models for the task of detection or classification can be seen in [1] for activity recognition, and [5] for object part detection. In our model, *chains* are used as a spatio-temporal detection, which localizes the discriminative part of the action.

We assign a score to each *chain*, and propose action detections as the chains with highest probabilities. This is a spatio-temporal generalization of methods for object detection in still images where multiple bounding boxes are proposed and scored based on their confidence. We propose *chains* as spatio-temporal bounding boxes that simultaneously localize and classify the action.

As opposed to the task of object-detection in still images, the localization of actions in videos is compounded by the fact that an action has to be localized both in the spatial dimension and the temporal one. Instead of attempting to localize the entire object performing the action, we aim at identifying the salient parts of the action. For instance, in the case of hand-waving our method will attempt to localize the hand motion in the sequence, rather than the entire person performing the action.

Since our chains are detected independently, our method has the added advantage of being invariant to the location of the action within the video. Therefore, we can detect multiple instances of an action type within a single video, or the simultaneous occurrence of different action types.

2. Related Work

Trajectories-based action recognition:
Trajectories have recently been used for the purpose for action recognition in various methods. Wang et al. [12] showed that representing trajectories with different types of features (HOG, HOF, and MBH) achieves good classification accuracy on various datasets. We use the same method for computing trajectories and the features related to them. However we use these features in a detection as well as classification task. Other relevant methods that use trajectories for video representation are [4, 7] and [9]. In [4], hierarchical clustering is performed on the trajectories, and the classification is performed using a kernel that computes the structural and visual similarity of two hierarchical decompositions. These methods rely on SVM classification and use a bag of words representation of features, which has been shown to suffer from quantization error. Our classification method uses the features directly without clustering them in a way which reduces their discriminative power.

In [14] and [10] the task of action recognition is formulated as a branch and bound search over subvolumes in the video. Each subvolume is scored according to the likelihood ratio of the features it contains. Our method also benefits from the discriminative scores of individual features, yet at the same time we make use of temporal relations between feature subsets.

Probabilistic chains-models: Chain models have been proposed for object part detection in [5]. The key idea of this work was in estimating the probability of pairs of features in the object and computing a probability for a chain of features as the product of the pair probabilities. We use the same approach of estimating chain probabilities, however our model constructs a spatio-temporal chain of grid points which contain multiple features.

Another work [1] proposes spatio-temporal chains for multiple person activity recognition in videos. Similarly to our work, they classify activities using chains of points in a spatio-temporal grid. Our work, however differs in that we use the computed chains directly as action detections and are thus able to localize each action both in the time and location dimensions. In addition, we rely directly on the computed trajectories without using body pose detections which are prone to error, and limit the use of the method to cases where the full body is visible.

Naive Bayes Nearest Neighbors (NBNN) [2] has been proposed as a method for classifying images based on the distance of features in the image to the different image categories. Despite its simplicity, this method has been shown to perform well for the image classification task. By foregoing the feature clustering step which is common to BOW based methods, the discriminative ability of the features is preserved. We apply this classification method on videos, and thus show that NBNN can be used for the task of action classification.

3. Method

Our action detection and classification method is divided into two parts. We first divide the video into a spatio-temporal grid in which each grid cell is a subvolume of the video. Next, we compute trajectory features for the video and assign them to different grid cells, according to their location in the video. We apply a Naive Bayes Nearest Neighbor (NBNN) classifier on each grid cell separately and assign a classification score to each grid cell. The motivation for using the NBNN classifier at this step is to overcome the quantization error caused by BOW representation. Next we combine neighboring grid cells into chains by estimating the joint probability of each pair of neighbors. In order to learn the probability distribution of pairs of grid cells we estimate a covariance matrix using a bag of words representation for each grid cell. In this way each grid cell is compactly represented as a histogram vector which can be used for estimating the parameters of the probabilistic model.

The different steps of the method are described in more detail in the following.

3.1. Features, Codebook and Grid cells

We compute trajectory features according the method of [12], which proposes the use of dense trajectories for video representation. Each trajectory is computed by tracking points on a grid using the optical flow field. In order to avoid drifting effects, the length of the trajectories is limited to 15 frames. Each trajectory is associated with a feature vector, which is computed within a space-time volume around the

trajectory. The feature vector contains the values of the histogram of oriented gradients (HOG), histogram of optical flow (HOF), motion boundary histogram (MBH), and the trajectory coordinate differences. These features have been used in various video classification methods and have been shown to perform well for discriminative models [14, 8].

Codebook generation: In order to obtain the histogram representation used for training the generative model, we first cluster the feature vectors into 4 separate dictionaries based on their types. We use a code book size of 300 and perform k-means clustering on the features from the training set. By assigning each feature vector to four different clusters we associate each feature with a unique codeword from a combined dictionary of size $4 \times 300 = 1200$. We found that this relatively small dictionary size achieves good performance, as will be described later. In order to reduce the dimensions of the histogram vectors, we project each histogram into a lower dimensional space of size 100 using PCA. The need for a low dimensional histogram representation will become evident in training phase, as described in the next section.

Next we divide the video sequence into uniform spatio-temporal grid, in which each grid cell is a three dimensional volume. Each feature is assigned to the grid cell with which it has the highest overlap. In this fashion we construct a histogram for each grid cell by binning the features associated with the cell according to the predefined codebook.

3.2. Training

Training the NBNN classifier:

Each grid cell is assigned a classification score by performing an ANN search for each feature contained in the grid cell, and summing the distances between all features to the different action classes. Therefore, training the NBNN classifier consists of extracting features from the training set videos and dividing the features into different classes based on the ground truth annotation of the video.

Training the generative model: We propose a generative model in which a covariance matrix is estimated from pairs of neighboring grid cells. Pairs of grid cells which are spatio-temporally adjacent are sampled from the training set, and their histograms are concatenated in a single vector:

$$H_{ij} = [h_i, h_j]^T \tag{1}$$

We select neighboring grid cells as follows: for each grid cell with coordinates (r, c, t), we set a neighborhood of 9 grid cells from the next time step $N = \{(r+k, c+l, t+1)\}_{k,l=\{-1,0,1\}}$.

A covariance matrix is computed from the set of all positively annotated grid pairs. This covariance matrix is used in the testing phase to evaluate the similarity between neighboring grid cells as a weighted distance function. In addition we compute the average histogram from the positive examples.

$$C = \frac{1}{N} \sum_{\substack{i,j \\ j \in N_i}} (H_{ij} - \mu) \cdot (H_{ij} - \mu)^T \tag{2}$$

where $\mu = [mean(h_i), mean(h_j)]^T$ is the average over all pairs of positive training histograms.

In this way we learn a class-specific model which describes the statistical dependence between pairs of neighboring cells.

3.3. Inference

Given a new video example, we perform the same initial steps as in the training phase: dividing the volume into a uniform grid, and computing a bag of words representation from each grid cell, and project the histograms into a lower dimension space using PCA. The NBNN classification is applied separately on each grid cell, and the class label and confidence score of each grid cell is stored. The unary probability associated with individual grid points was computed according to:

$$P(i|A) = \begin{cases} \exp(-\frac{d_A}{d_{min2}}) & \text{if } A = argmin_c\{d_c\} \\ 0 & o.w. \end{cases} \tag{3}$$

where d_c is the sum of feature distances to the class c, and d_{min2} is the sum of distances to the class with the second smallest distances.

Next we extend each grid cell into a chain that attempts to link grid cells which correspond to the learned generative model. We define a distance between a pair of histograms as the covariance weighted mahalonobis distance. The covariance matrix which was computed on the training data stores information about the co-occurrence of neighboring grid cells. The similarities between different dimensions of the histograms are weighted by their covariance. This is equivalent to assuming a joint gaussian distribution between pairs of neighboring grid cells, and estimating the probability of a pair of histograms based on the estimated gaussian distribution. The probability under this assumption is computed using the learned model parameters C and μ:

$$d(h_i, h_j) = (H_{ij} - \mu)^T \cdot C^{-1} \cdot (H_{ij} - \mu) \tag{4}$$

$$P(i, j) = \frac{1}{K} exp(-d(h_i, h_j)) \tag{5}$$

Where $H_{ij} = [h_i, h_j]^T$ for histograms h_i, h_j of two neighboring grid cells, and K is a normalization constant.

Chain formation: In order to extend chains for action detection we evaluate the above probability for each pair of neighboring grid cells. Our goal is to form chains of

a given length (L_{chain}), which minimize the distances between neighboring grid cells. We can achieve this by defining a graph using the grid cells as nodes, and the neighborhood relations as the edges, and find optimal paths of length L_{chain} using dynamic programming, as described in alg. 1. In this way we form optimal chains, rather than connecting grid cells in a greedy fashion.

Next we assign a confidence score to each chain. Assuming that a given chain is used as a detection for action class A, the confidence score for this detection is computed as the probability of detection, using the unary and pairwise probabilities previously computed:

$$P_{chain|A} = \prod_{i=1}^{L_{chain}} P_i \cdot P_{i,i+1} \tag{6}$$

where P_i, and $P_{i,i+1}$ are the unary and pairwise probabilities defined in eqn. 3 and 5 respectively.

Algorithm 1 Input: d = distance between each pair of grid points; Idx = grid index set at time t; Output: $chain\_link$ = the chain edges between pairs of grid points.

procedure CHAINFORM($d(i,j), Idx$)
  $Accum\,(:) \leftarrow 0$
  $t \leftarrow T_{max}$
  while $t > 0$ **do**
    for $i \in Idx\{t\}$ **do**
      $d_s \leftarrow min_{j \in N_i} d\,(i,j)$
      $j_s \leftarrow argmin_{j \in N_i} d\,(i,j)$
      $Accum\,(i) \leftarrow Accum\,(j_s) + d_s$
    end for
    $t \leftarrow t - 1$
  end while
  $t \leftarrow 0$
  while $t < T_{max}$ **do**
    for $i \in Idx\{t\}$ **do**
      $j_s \leftarrow argmin_{j \in N_i} Accum\,(j)$
      $chain\_link\,(i) \leftarrow j_s$
    end for
    $t \leftarrow t + 1$
  end while
  return $chain\_link$
end procedure

Non-maxima suppression: Since we form chains of length L_{chain} starting from each grid cell, we obtain a set of chains which are highly overlapping. In order to reduce the number of detections (chains) we perform a non-maxima suppression step on the set. The goal here is to find a subset of non-overlapping chains which have maximal scores. We therefore keep only the chains with the maximal score from all groups of overlapping chains from different classes. We remove non-maximal chains which share the same temporal as well as spatial region with other chains in the video.

4. Experimental Results

We evaluate our method on a challenging action recognition dataset. As opposed to other methods, we are able to localize the action in each video according to its spatial position and temporal occurrence within the video, and therefore we evaluate our method based on localization as well as classification performance.

The dataset on which we evaluate our model is the Cooking Activity dataset [8]. This dataset contains videos of daily actions performed in a natural setting. Each video in the dataset contains multiple action classes performed by different subjects. The differences between the actions are often very subtle, therefore making the classification a challenging task.

We split the dataset into two subsets, one used for training and the other for evaluating performance. Following the evalution procedure of [8], we used videos from different subjects for training and for testing. We trained our model on 7 subjects, while the remaining 5 subjects were used for testing. The codebook, covariance matrix, and NBNN classifiers are learned on these training set videos. We compare our results to the results reported by [8].

As opposed to [8], we resize the video frames and reduce their resolution from 1624×1224 to 480×360 pixels. We then compute trajectory features on the resized videos. As a result, we compute about 16 times less trajectories per video frame. While reducing the total number of trajectories, this step allows us to maintain low memory requirements for constructing our model, and speed up the ANN search for the NBNN classification. We show that despite the loss of the information available in the full video, we still achieve competitive performance.

For evaluation, we compute precision-recall curves by summing the number of detections for all the videos in the dataset. A chain is considered a true positive detection if its assigned label matches the label of the ground truth annotation with which it intersects. We use the following threshold criterion for counting true positives (we count a detection as true positive if half of its volume is within the ground truth annotation).

We set a threshold for the detection scores and compute true positives and false detections for the whole dataset. By varying the threshold score, we obtain a precision recall curve for varying number of detections (fig. 3). In this figure we report results from chains of different lengths, and compare them to the detection result obtained in [8]. As can be seen in this figure, we achieve comparable results to the state of the art (labeled as "Temporal Window" in the graph).

Our method has the added advantage of being able to localize the discriminative part of the action within each

frame of the video. In fig. 4 we show example detection and classification results for various action classes. As can be seen in this figure, the discriminative part of the action (e.g. the hand motion) is detected by the chains model. We show results for 2 sizes of the spatio-temporal grid. The smaller grid size captures less features from the action and thus achieves lower performance, yet is better able to localize the discriminative part of the action.

In addition we experiment with different grid sizes. Detection results for the different grid size parameter can be seen in figure 4. As expected, using a larger grid size improves the classification performance, since each grid cell contains more features.

Finally, we report the average precision (AP) for each individual action class in table 1. It is evident from this table that for action classes that have a small number of examples in the dataset, our chain model results in low performance , while for other categories with a larger number of training examples our method achieves higher performance.

The mean AP for all action classes is 23% for our chains model. This is lower than the result reported by [8] (45%), yet we use a smaller number of features, and perform a harder task of localizing the action within the frame.

We evaluate the performance of our method on a second action recognition dataset - the MSR-A dataset [14]. This dataset contains challenging scenes, where background clutter makes the recognition task difficult. Similar to [3] and [14], we trained our model using the KTH dataset which contains similar action classes, and evaluate our performance on the MSR-A data. The three classes which we try to recognize are: hand-waving, hand-clapping, and boxing.

Fig. 6 shows results of our method for the three action classes under this setting. The mean AP of the three classes is 27.4%. In Fig. 7 we show the performance of our method using chains formed without computing the mahalonobis distance between neighbors. Instead, the chains were created by selecting a random neighbor for each grid cell. As expected, the performance drops under this setting (mAP = 19.3%), demonstrating the importance of the generative component in our model.

Example detections for both datasets are shown in figures 4 and 5 respectively.

5. Conclusions and future work

In this paper we have proposed a method for human action detection by detecting chains of grid points in a spatio-temporal representation of the video. The chains are constructed using dynamic programming approach which links points into chains with minimal cost, as learned by a probabilistic representation of the action classes.

We show that by modeling probability of pairs of features according to their temporal relation in the video, we

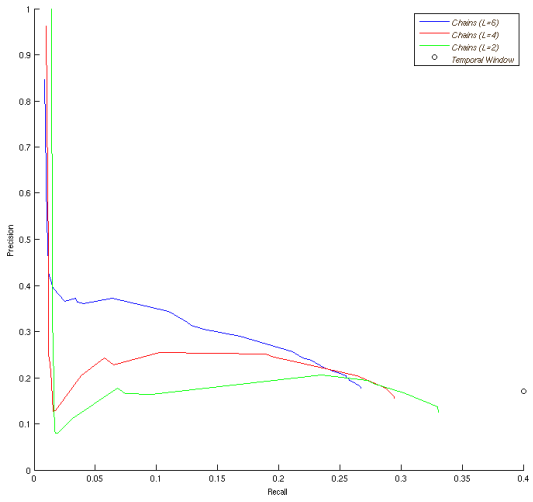

Figure 3. Precision-recall curve for action classes from the Kitchen Activity action dataset. The curves were created by varying the threshold score for the chains detections in the whole dataset. We compare chains of different lengths, using a small grid size of $50 \times 50 \times 15$.

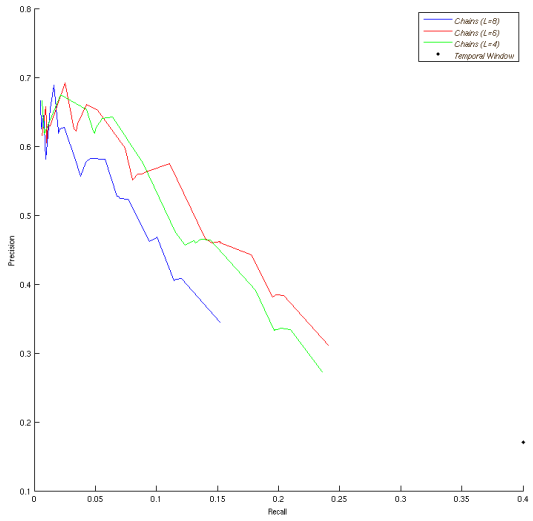

Figure 4. Precision-recall curve for action classes from the Kitchen Activity action dataset. The curves were created by varying the threshold score for the chains detections in the whole dataset. We compare chains of different lengths, using a larger grid size of $100 \times 100 \times 20$.

are able to extend the standard bounding box detections into spatio-temporal regions that capture the meaningful and discriminative parts of the action. Furthermore, we demonstrated that by reducing the size of the grid, we are able to achieve a fine-grained localization of the disciminative

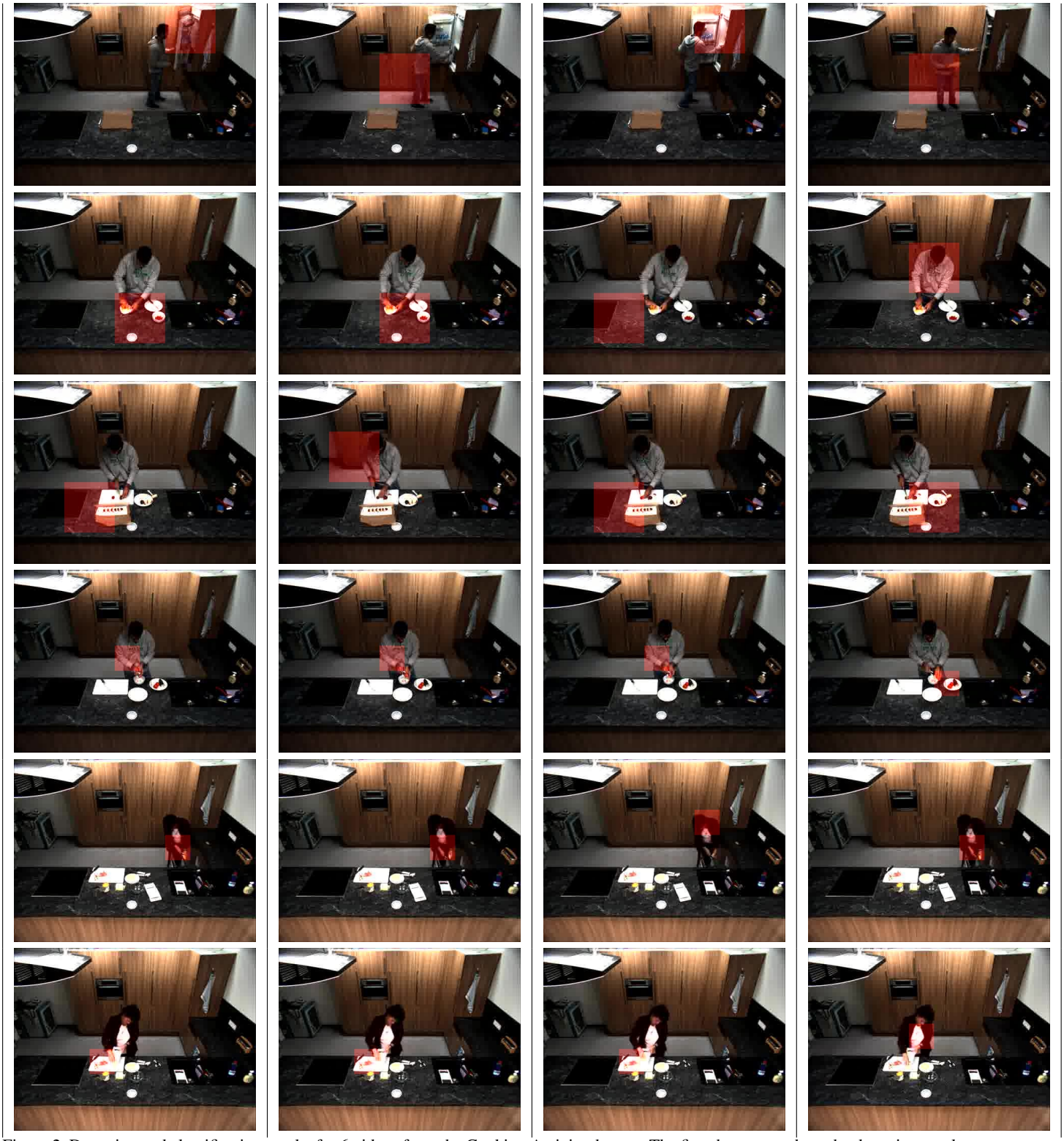

Figure 2. Detection and classification results for 6 videos from the Cooking Activity dataset. The first three rows show the detection results for "take out from fridge", "squeeze" and "cut slices" actions using a grid size of $100 \times 100 \times 20$ pixels. The bottom 3 rows show results for "peeling" and "grating" actions using a smaller grid size of $50 \times 50 \times 15$.

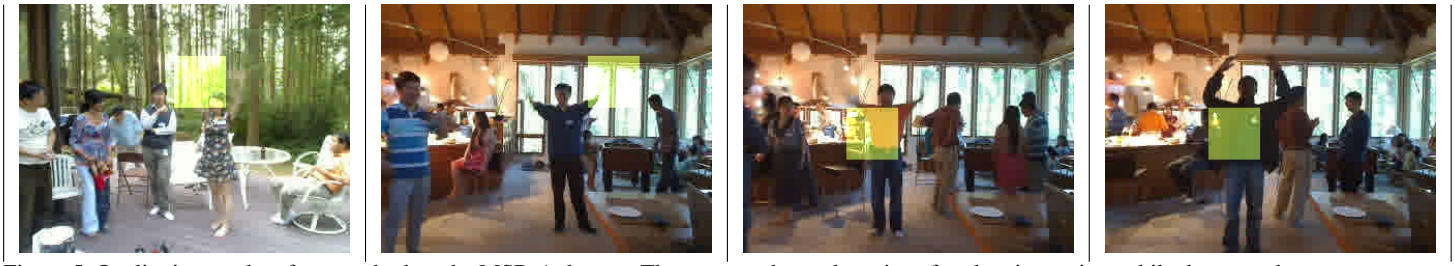

Figure 5. Qualitative results of our method on the MSR-A dataset. The top row shows detections for clapping action, while the second row show detections for hand waving. The detections displayed above were the ones that were assigned the highest confidence score (see text)

Action class	[8]	Chains	Action class	[8]	Chains
Background activity	**47.1**	42.96	put on plate	11.0	**16.36**
change temperature	**37.6**	32.22	read	**34.5**	27.77
cut apart	**16.0**	03.53	remove from package	**39.1**	11.90
cut dice	**25.1**	1.63	rip open	**5.8**	0.0
cut in	22.8	**25.0**	scratch off	**3.8**	0.0
cut off ends	7.4	**18.0**	screw close	**36.3**	24.21
cut out inside	16.3	**19.48**	screw open	19.1	**48.43**
cut slices	**42.0**	0.37	shake	**33.5**	05.20
cut stripes	**27.6**	11.11	smell	**24.8**	0.0
dry	**95.5**	10.81	spice	**29.3**	03.17
fill water from tap	75.0	75.0	spread	11.2	**30.76**
grate	**32.9**	1.07	squeeze	**90.0**	01.45
lid: put on	2.0	**90.90**	stamp	**73.3**	13.88
lid: remove	1.9	**93.33**	stir	**50.0**	00.36
mix	36.8	**50.0**	strew	39.6	**48.64**
move from x to y	**15.9**	7.04	take & put in cupboard	37.2	**44.44**
open egg	**45.2**	21.42	take & put in drawer	**37.6**	0.0
open tin	**79.5**	5.14	take & put in fridge	**54.6**	07.81
open/close cupboard	**54.0**	29.62	take & put in oven	100.	**80.0**
open/close drawer	**38.1**	03.89	t. & put in spice holder	80.2	**85.71**
open/close fridge	**73.7**	37.5	take ingredient apart	**17.5**	12.01
open/close oven	**25.0**	0	take out from cupboard	**81.5**	00.28
package x	**31.9**	0	take out from drawer	**79.7**	0.0
peel	**65.2**	0.64	take out from fridge	**73.6**	0.62
plug in/out	**54.7**	0.0	take out from oven	**83.3**	75.0
pour	**54.2**	06.96	t. out from spice holder	**67.0**	30.0
pull out	**87.5**	0.0	taste	**18.2**	0.0
puree	**67.1**	07.5	throw in garbage	**84.4**	07.02
put in bowl	**18.8**	1.83	unroll dough	**100.**	0.0
put in pan/pot	**15.3**	08.86	wash hands	**45.9**	07.28
put on bread/dough	**42.1**	03.97	wash objects	**67.1**	00.71
put on cutting-board	**7.1**	0.0	whisk	**70.0**	09.72
wipe clean	**10.6**	0			

Table 1. AP results for different action classes from the Cooking activity dataset, comparison to [8]

parts, at the cost of reducing the accuracy over the dataset.

As future work, we intend to improve the classification performance of our model by using the NBNN classifier to learn a kernel which is used by an SVM for the classification task. This approach has been described in [11]. Using this approach would allow us to combine the different features we use for the video representation in a discriminative way.

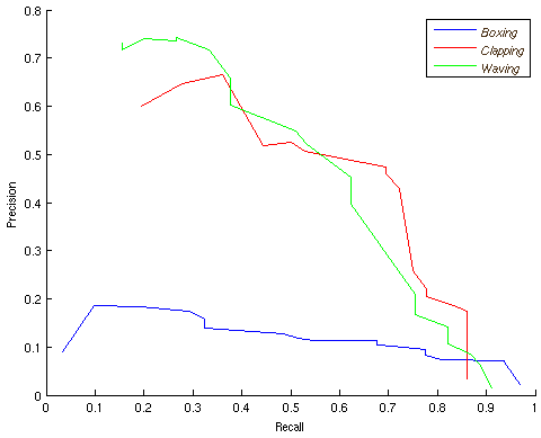

Figure 6. Precision-recall curve for 3 action classes from the MSR action dataset. The curves were obtained by varying the detection score threshold.

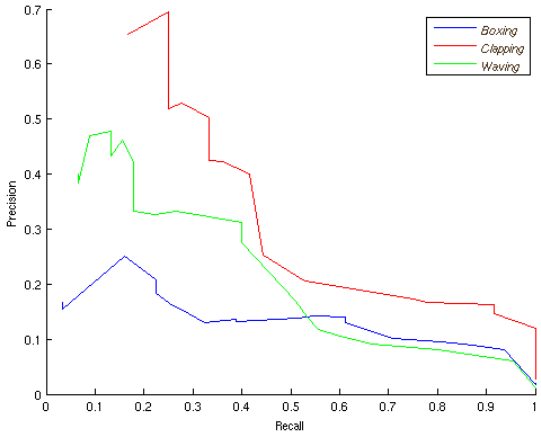

Figure 7. Precision-recall curve for 3 action classes from the MSR action dataset, with random chain formation. The chains in this experiment were formed by taking random neighbors for each chain.

Acknowledgments

This work was supported in part by the KU Leuven OT project VASI and FWO project "Monitoring of abnormal activity with camera systems".

References

[1] M. Amer and S. Todorovic. A chains model for localizing participants of group activities in videos. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 786–793, 2011. 1, 2

[2] O. Boiman, E. Shechtman, and M. Irani. In Defence of Nearest-Neighbor ased Image Classification. In *IEEE Conference on Computer Vision & Pattern Recognition*, June 2008. 2

[3] L. Cao, Z. Liu, and T. Huang. Cross-dataset action detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1998–2005, 2010. 5

[4] A. Gaidon, Z. Harchaoui, and C. Schmid. Recognizing activities with cluster-trees of tracklets. In *BMVC*, Guildford, Royaume-Uni, Sept. 2012. 2

[5] L. Karlinsky, M. Dinerstein, D. Harari, and S. Ullman. The chains model for detecting parts by their context. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 25–32, 2010. 1, 2

[6] J. C. Niebles, H. Wang, and L. Fei-fei. Unsupervised learning of human action categories using spatial-temporal words. In *In Proc. BMVC*, 2006. 1

[7] M. Raptis, I. Kokkinos, and S. Soatto. Discovering discriminative action parts from mid-level video representations. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, United States, June 2012. IEEE, IEEE. 2

[8] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, United States, June 2012. IEEE, IEEE. The dataset and relevant code is available at http://www.d2.mpi-inf.mpg.de/mpii-cooking. 3, 4, 5, 7

[9] M. Sapienza, F. Cuzzolin, and P. Torr. Learning discriminative space-time actions from weakly labelled videos. In *2012 BMVC*, 2012. 2

[10] Y. Tian, L. Cao, Z. Liu, and Z. Zhang. Hierarchical filtered motion for action recognition in crowded videos. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(3):313–323, 2012. 2

[11] T. Tuytelaars, M. Fritz, K. Saenko, and T. Darrell. The nbnn kernel. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1824–1831, 2011. 7

[12] H. Wang, A. Kläser, C. Schmid, and L. Cheng-Lin. Action Recognition by Dense Trajectories. In *IEEE Conference on Computer Vision & Pattern Recognition*, pages 3169–3176, Colorado Springs, United States, June 2011. MSR - INRIA. 1, 2

[13] Y. Wang and G. Mori. Human action recognition by semi-latent topic models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(10):1762–1774, 2009. 1

[14] J. Yuan, Z. Liu, and Y. Wu. Discriminative video pattern search for efficient action detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(9):1728–1743, Sept. 2011. 1, 2, 3, 5